



# VEGA PROCESSORS



## Digital India RISC-V VEGA Processor (Fully indigenous RISC-V Processor)



# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Installing VEGA SDK

# Installing VEGA SDK



## What you will learn?

- Install vega-tools
- Install vega-sdk
- Building a project
- Running Program on Target Board
- Creating a New Project
- Interrupt-Based Example
- Using SDK in IDE (Eclipse)
- Executing the Program on Target Board via Ethernet



# Installing VEGA SDK



## Supported Systems

- Linux machines only
- Ubuntu



# Supported Systems

Currently the Linux distributions that support VEGA SDK are :

- Ubuntu 18.04
- Ubuntu 20.04
- Ubuntu 22.04
- Ubuntu 24.04
- We expect other Linux distributions to work as well, provided that the user can either run our provided toolchains on them or produce their own toolchain.

# Prerequisites



To use this SDK, you will need the following software available on your Linux machine :

## GNU Make

- sudo apt update
- sudo apt install build-essential

## GNU Autoconf

- sudo apt install autoconf

## Git

- sudo apt install git

## Minicom

- sudo apt install minicom



```
~/VEGA$ sudo apt update
~/VEGA$ sudo apt install build-essential
~/VEGA$ sudo apt install autoconf
~/VEGA$ sudo apt install git
~/VEGA$ sudo apt install minicom
```

# Install VEGA Tools



- VEGA Tools contains necessary RISC-V Toolchains to compile C/C++ programs and other utilities.
- To Install VEGA Tools :

## For TRISUL32 (Arty A7-100T)

- git clone https://gitlab.com/dirv-vega-gc2025/vega-tools.git
- cd vega-tools
- ./setup-env.sh
- cd ..

```
~VEGA$ git clone https://gitlab.com/dirv-vega-gc2025/vega-tools.git
~VEGA$ cd vega-tools
~VEGA/vega-tools$ ./setup-env.sh
~VEGA/vega-tools$ cd ..
```

# Install VEGA SDK



## Clone and install VEGA SDK

- git clone https://gitlab.com/dirv-vega-gc2025/vega-sdk.git
- cd vega-sdk



```
~VEGA$ git clone https://gitlab.com/dirv-vega-gc2025/vega-sdk.git  
~VEGA$ cd vega-sdk
```

# Install VEGA SDK (contd...)



VEGA SDK supports several development boards.

Choose the setup script according to your target development board.

## For TRISUL32 (Arty A7-100T)

- ./setup-trisul32.sh

If all steps are successful,  
you'll see the following output  
after running

./setup-trisul32.sh

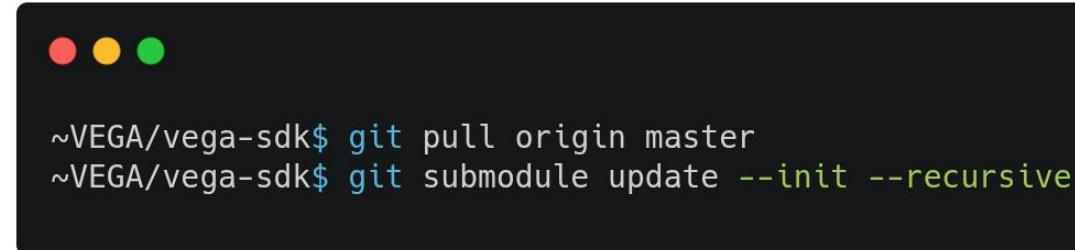
```
~/VEGA/vega-sdk$ ./setup-trisul32.sh
```

```
Setting up VEGA SDK Environment
configure.ac:6: installing './ar-lib'
configure.ac:3: installing './compile'
configure.ac:2: installing './install-sh'
configure.ac:2: installing './missing'
Makefile.am: installing './depcomp'
Configuring minicom & adding user to 'dialout' group
[sudo] password for hdg:
-----
Minicom configured for thejas32 hardware & user added to 'dialout' group.
Command for uart access: minicom aries
[If permission denied, try restart the system]
-----
VEGA SDK Environment added
```

# Updating the SDK/Tools

If you'd like to update your SDK/Tools to the latest version, go to respective directory and :

- git pull origin master
- git submodule update --init --recursive

A screenshot of a macOS terminal window. The window has three colored title bar buttons (red, yellow, green) at the top. The main area contains two lines of text:  
~VEGA/vega-sdk\$ git pull origin master  
~VEGA/vega-sdk\$ git submodule update --init --recursive

## Note

- It is recommended to update VEGA Tools before you update VEGA SDK.

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Building a Project

# Building a Project

To compile an example project, say **hello\_world**

- cd examples/uart/hello\_world/
- make

If your make command end like following,

```
Build successful!
ELF : hello
Binary : hello.bin
Hex : hello.hex
Dump   : hello.dump
Files are generated in build folder.
Size information
    text      data      bss      dec      hex filename
  26072        36       84     26192      6650 build/hello.elf
```

- Then your build is successful.
- After a successful build, an **elf** file, **.bin** file, **.dump** file and a **.hex** file will be created in the **build/** directory.
- **.dump** file contains your program in assembly form.

```
~/VEGA/vega-sdk$ cd examples/uart/hello_world/
~/VEGA/vega-sdk/examples/uart/hello_world$ make
```

# Cleaning a Target Project Build Directory



The clean target can be used to restore a target project's directory to a clean state.

- make clean

This will remove the **build/ directory** of the target project.



```
~/VEGA/vega-sdk/examples/uart/hello_world$ make clean
```

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Running Program on Target Board

# Running Program on Target Board

## Setting Up Serial Device

- We are using Linux application minicom for serial communication.
- You could use any other serial communication applications like **python-serial**.

## Note

- Make sure that the board is connected to the host PC via micro USB cable.
- Also ensure that you use the correct serial device! In most cases it would be /dev/ttyUSB1.
- `minicom -h` will give you an idea of options available in minicom.



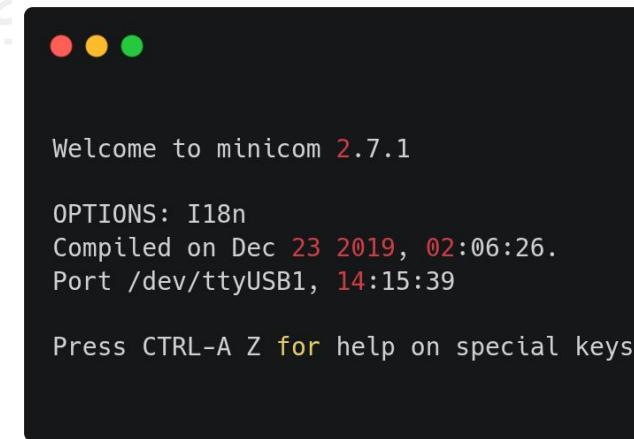
# Running Program on Target Board

## To setup minicom :

- sudo minicom trisul32



```
~VEGA$ sudo minicom trisul32
```



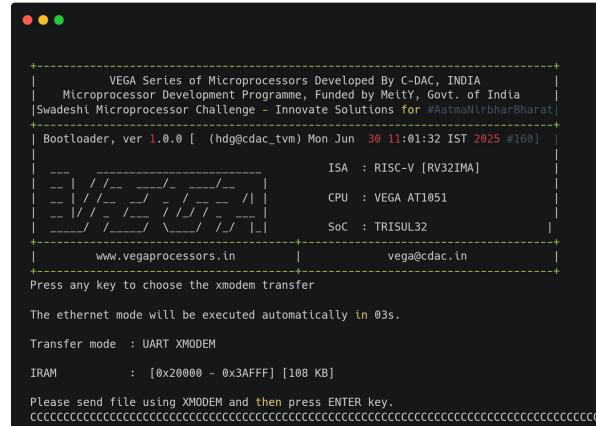
```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB1, 14:15:39

Press CTRL-A Z for help on special keys
```

# Resetting Target Board

- You could reset target board by pressing the reset button.
- You should reset the Board every time before you upload programs to the Board.
- You could find reset button of your target board from [Board pin mapping](#).
- If your reset was correct you will see something like this in terminal :



# Uploading Program to Target Board



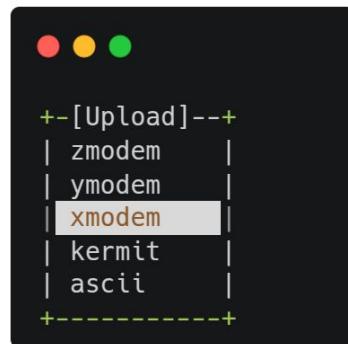
- While uploading, we are transferring the **.bin** file to the target board and executing that file on target board.
- VEGA SDK supports many methods for uploading program.
- Follow the method supported by your target board.

<b>TRISUL32 (Arty 100T)</b>	Uploading Using UART
	Uploading Using Ethernet

# Uploading Using UART

Uploading using UART method uses XMODEM protocol to upload program.

- Open Minicom and press the RESET button to restart the board.
- Press any key to choose the xmodem transfer and make sure Transfer mode is UART XMODEM.
- Use **CTRL+A S** to enter file sending menu and select xmodem by pressing Enter.



# Uploading Using UART

(contd...)



- In the next window, with space bar select the `.bin` file to be transferred, by pressing Enter, transfer process starts.
- Wait until the process is completed. The screen should display how much data has been transferred.

```
+[Select a file for upload]
|Directory: /home/DATA2/VEGA/vega-sdk/bin
|[...]
|hello.bin
|
|
|
| ( Escape to exit, Space to tag )
+-----+
```

[Goto] [Prev] [Show] [Tag] [Untag] [Okay]

```
+-----[xmodem upload - Press CTRL-C to quit]-----+
|Sending hello.bin, 203 blocks: Give your local XMODEM receive|
| command now.
|Bytes Sent: 26112    BPS:7783
|
|Transfer complete
|
|READY: press any key to continue...
+-----+
```

# Uploading Using UART (contd...)

- After completing transfer, the Program will start to execute.



```
Transfer mode : UART XMODEM

IRAM       : [0x20000 - 0x3AFFF] [108 KB]

Please send file using XMODEM and then press ENTER key.

CCCCCCCC

Hello World

EXIT
```

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Creating and Uploading a New Project to the Target  
Board

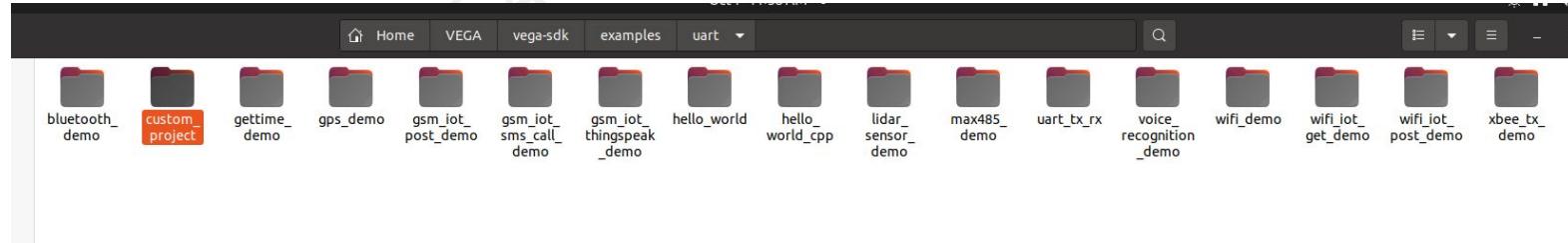
# Creating a Project

It is recommended to create a new project by copying any sample projects from **examples/** directory.

You can also make a new Project by just copying the MakeFile only.

## Recommended Method :

- Copy any example project and rename, say `custom_project`.



# Creating a Project

(contd...)

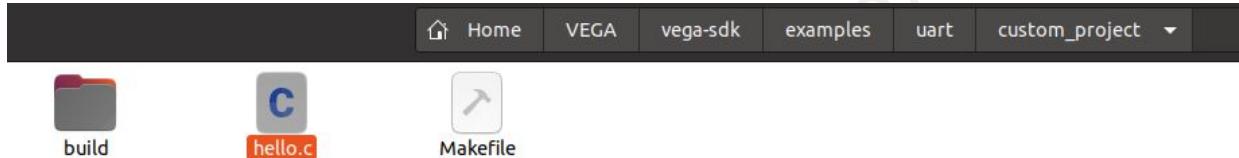
- Change the EXECUTABLE\_NAME variable in the MakeFile to  
EXECUTABLE\_NAME=  
custom\_project.

```
1 #----- : MDP - Microprocessor Development Project
2 #Project Name      : HD083D
3 #Project Code       : 07-Jan-2020
4 #Created           : Makefile
5 #Filename          : Sample hello world program
6 #Purpose            : Sample hello program with uart print
7 #Description        : Premjith A V
8 #Author(s)          : premjith@cdac.in
9 #Email              : -----
10 #----- : See LICENSE for license details.
11 #----- : -----
12 #----- : -----
13 ##### : -----
14 # Configurations : -----
15 ##### : -----
16 # Include the BSP settings : -----
17
18 CONFIG_PATH=~/config/vega-tools/settings.mk
19 ifeq ("$(wildcard ${CONFIG_PATH})", "")
20 ${error} Please install [VEGA SDK]/[VEGA Tools] and setup the environment)
21 endif
22
23 include ${CONFIG_PATH}
24
25 ifeq ("$(wildcard ${VEGA_TOOLCHAIN_PATH})", "")
26 ${error} Please install [VEGA Tools] and setup the environment)
27 endif
28
29 ifeq ("$(wildcard ${VEGA_SDK})", "")
30 ${error} Please install [VEGA SDK] and setup the environment)
31 endif
32 SDK_PATH=${VEGA_SDK}
33
34 ##### : -----
35 # Executable name : -----
36 ##### : -----
37 EXECUTABLE_NAME=custom_project
38
39
40 include ${SDK_PATH}/bsp/common/config.mk
41
```

# Creating a Project

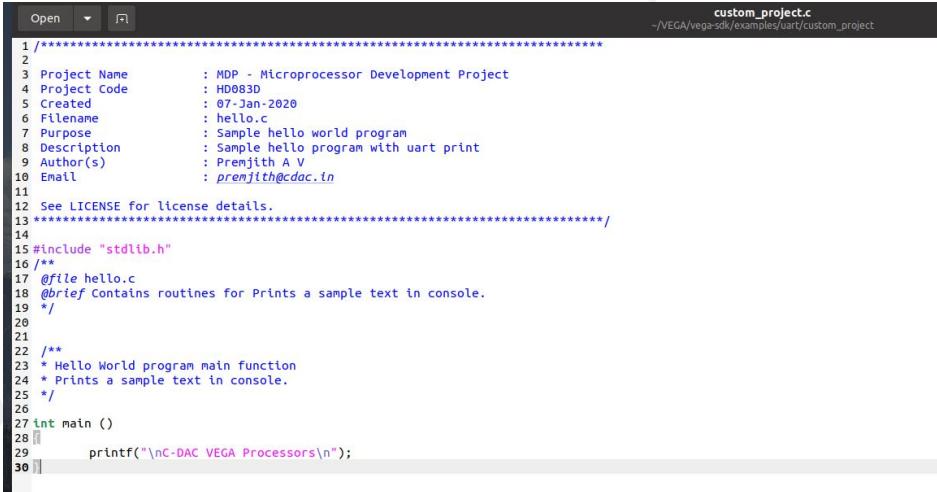
(contd...)

- Remove all .c and .h files in custom\_project.



# Creating a Project (contd...)

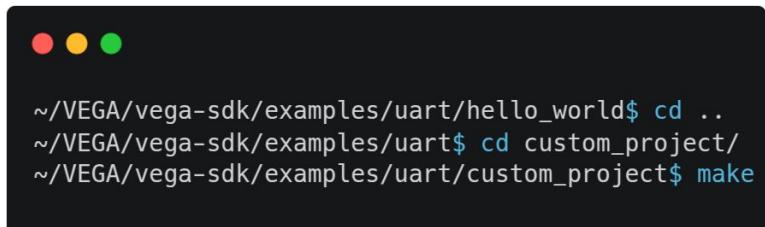
- Create your own .c file say, custom\_project.c and implement your logic.



```
custom_project.c
~/VEGA/vega-sdk/examples/dart/custom_project

1 ****
2
3 Project Name      : MDP - Microprocessor Development Project
4 Project Code       : HD083D
5 Created            : 07-Jan-2020
6 Filename           : hello.c
7 Purpose             : Sample hello world program
8 Description         : Sample hello program with uart print
9 Author(s)          : Premjith A V
10 Email              : premjith@cdac.in
11
12 See LICENSE for license details.
13 ****
14
15 #include "stdlib.h"
16 /**
17 @file hello.c
18 @brief Contains routines for Prints a sample text in console.
19 */
20
21
22 /**
23 * Hello World program main function
24 * Prints a sample text in console.
25 */
26
27 int main ()
28 {
29     printf("\nC-DAC VEGA Processors\n");
30 }
```

- Use make command to build your project.



```
~/VEGA/vega-sdk/examples/uart/hello_world$ cd ..
~/VEGA/vega-sdk/examples/uart$ cd custom_project/
~/VEGA/vega-sdk/examples/uart/custom_project$ make
```

# Creating a Project

(contd...)

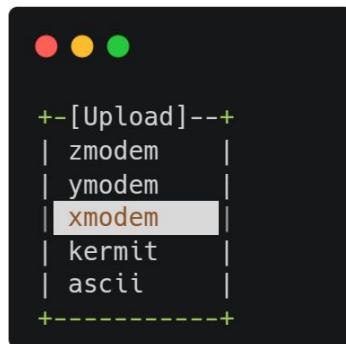
- If your make command end like following, Then your build is successful.

```
Build successful!
ELF : custom_project
Binary : custom_project.bin
Hex : custom_project.hex
Dump   : custom_project.dump
Files are generated in build folder.
Size information
      text     data     bss     dec     hex filename
    26812       36      84    26932     6934 build/custom_project.elf
```

# Uploading Using UART

Uploading using UART method uses XMODEM protocol to upload program.

- Open Minicom and press the **RESET** button to restart the board.
- Press any key to choose the xmodem transfer and make sure Transfer mode is UART XMODEM.
- Use **CTRL+A s** to enter file sending menu and select xmodem by pressing Enter.



# Uploading Using UART (contd...)

- In the next window, with space bar select the **.bin** file to be transferred, by pressing Enter, transfer process starts.
  - Wait until the process is completed. The screen should display how much data has been transferred.

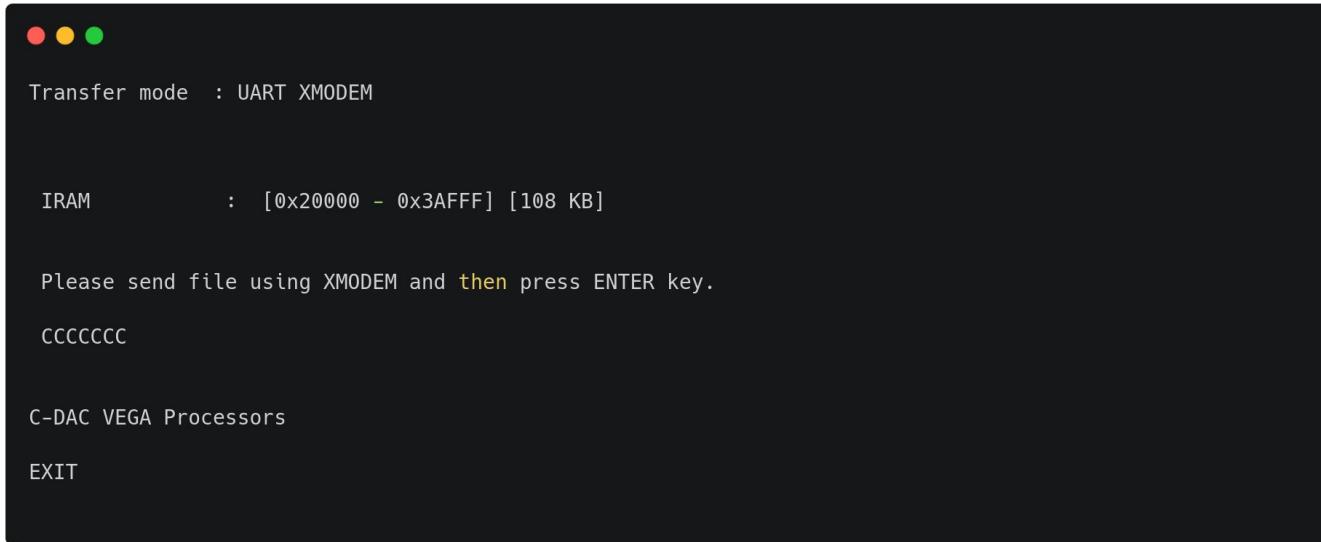
```
+-----[Select a file for upload]-----+
|Directory: /home/hdg/VEGA/vega-sdk/bin
|[...]
|custom_project.bn
|hello.bn

+-----( Escape to exit, Space to tag )-----+
[Goto] [Prev] [Show] [Tag] [Untag] [Okay]
```

```
[xmodem upload - Press CTRL-C to quit]
| Sending hello.bin, 203 blocks: Give your local XMODEM receive
| command now.
| Bytes Sent: 26112    BPS:7783
|
| Transfer complete
|
| READY: press any key to continue...
+
```

# Uploading Using UART (contd...)

- After completing transfer, the Program will start to execute.



```
Transfer mode : UART XMODEM

IRAM      : [0x20000 - 0x3AFFF] [108 KB]

Please send file using XMODEM and then press ENTER key.

CCCCCCC

C-DAC VEGA Processors

EXIT
```

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Interrupt-Based Example

# Example using interrupts



- Open the example **vega-sdk -> examples -> interrupt -> uart\_interrupt**
- Open the **uart\_interrupt.c** file
- Inside the main function, update the board frequency to match your board's clock.
- For instance, the TRISUL32 SoC used in this setup runs at 40 MHz.

```
#define UART0_INTR_NUM 1 // Note : this number is for thejas64, number may change on other SoCs

void uart_intr_handler(void) // interrupt handler
{
    unsigned char status, error, data;

    status = UartReg(UART_0).UART_IIR_FCR; // clearing interrupt
    data = uart_getchar(UART_0, &error); // reading data
    uart_putchar(UART_0, data, &error); // echoing data
    return;
}

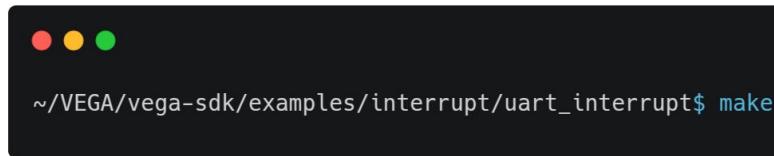
/**
@fn main
@brief transmit and reception through uart
@details
@param[in] No input parameter.
@param[out] No output parameter.
*/
void main()
{
    uart_set_baud_rate(UART_0, 115200, 4000000); // initializing uart0
    printf("-----[ uart%d in interrupt mode ]-----\n", UART_0);

    volatile unsigned int * Bdram = (unsigned int *) (0x20030000);
    * Bdram = (unsigned int) 0x00200002;

    interrupt_enable(UART0_INTR_NUM); //Enable interrupt in controller.
    irq_register_handler(UART0_INTR_NUM, uart_intr_handler);
    uart_intr_enable(UART_0, TX_DISABLE, RX_ENABLE); // enabling uart0 interrupt
    printf("\nStart typing anything on uart%d.\n\n", UART_0);
    /* idle loop */
    while (1);
}
```

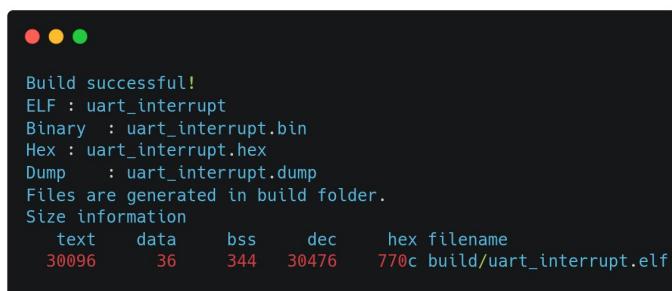
# Example using interrupts (contd...)

- Compile the project, say **uart\_interrupt**



```
~/VEGA/vega-sdk/examples/interrupt/uart_interrupt$ make
```

- If your make command end like following,then your build is successful.



```
Build successful!
ELF : uart_interrupt
Binary : uart_interrupt.bin
Hex : uart_interrupt.hex
Dump : uart_interrupt.dump
Files are generated in build folder.
Size information
    text      data      bss      dec      hex filename
 30096        36     344    30476    770c build/uart_interrupt.elf
```

# Example using interrupts (contd...)

- Open minicom :

```
~VEGA/vega-sdk/examples/interrupt/uart_interrupt$ sudo minicom trisul32
```

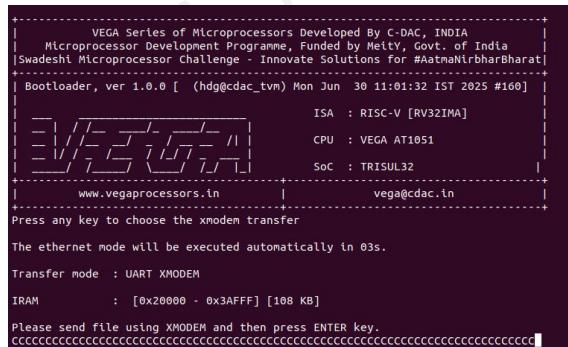
```
/VEGA/vega-sdk/examples/interrupt/uart_interrupt$ sudo minicom trisul32
[sudo] password for xyz:

Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB1, 14:42:08

Press CTRL-A Z for help on special keys
```

- Reset board and make sure Transfer mode is UART XMODEM.



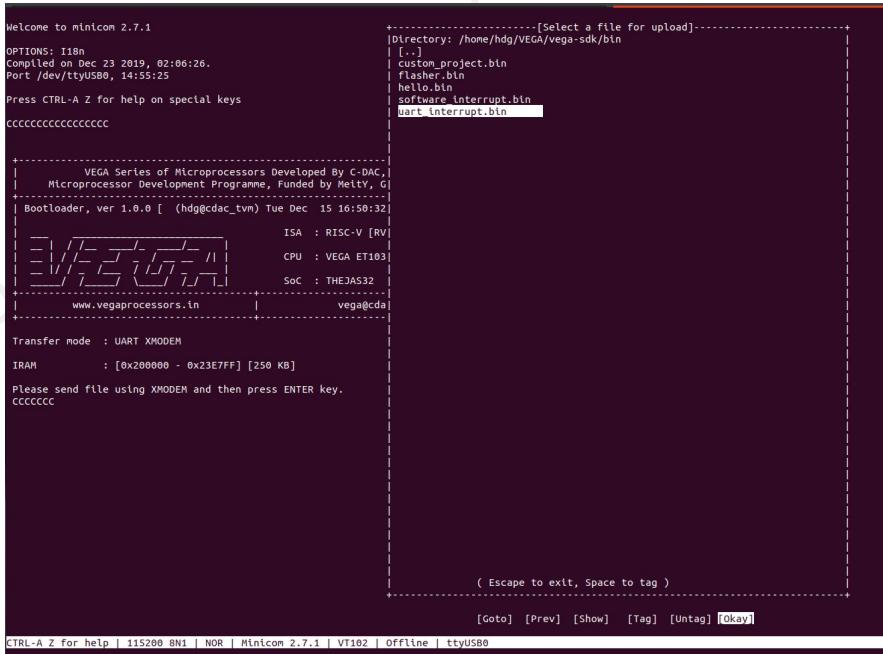
# Example using interrupts (contd...)

- Use **CTRL+A S** to enter file sending menu and select **xmodem** by pressing Enter.

```
Welcome to minicom 2.7.1
OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 14:55:25
Press CTRL-A Z for help on spi zmodem | ymodem |
CCCCCCCCCCCCCCCC | xmodem | kermit |
| ascii |
+-----+
| VEGA Series of Microprocessors Developed By C-DAC, INDIA |
| Microprocessor Development Programme, Funded by MeitY, Govt. of India |
+-----+
| Bootloader, ver 1.0.0 [ (hdg@cdac_tvm) Tue Dec 15 16:50:32 IST 2020 #135] |
| ISA : RISC-V [RV32IM] |
| CPU : VEGA ET1031 |
| SoC : THEJAS32 |
+-----+
| www.vegaprocessors.in | vega@cdac.in |
+-----+
Transfer mode : UART XMODEM
IRAM : [0x200000 - 0x23E7FF] [250 KB]
Please send file using XMODEM and then press ENTER key.
CCCCCCCC
```

# Example using interrupts (contd...)

- In the next window, with space bar, select the **.bin** file to be transferred, by pressing **Enter**, transfer process starts.



```
Welcome to minicom 2.7.1
OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 14:55:25
Press CTRL-A Z for help on special keys
CCCCCCCCCCCCCCCC

+-----[Select a file for upload]-----+
|Directory: /home/hdg/VEGA/vega-sdk/bin
| [...]
| custom_project.bin
| flasher.bin
| hello.bin
| software_interrupt.bin
| software_interrupt.bin
| uart_interrupt.bin

+-----[System Configuration]-----+
| VEGA Series of Microprocessors Developed By C-DAC,
| Microprocessor Development Programme, Funded by MeITY, G
| Bootloader, ver 1.0.0 [ (hdg@cdac_tvm) Tue Dec 15 16:50:32
| ISA : RISC-V [RV]
| CPU : VEGA ET103
| SoC : THEJA532
| www.vegaprocessors.ln | vega@cd
+-----[Transfer Mode]-----+
Transfer mode : UART XMODEM
IRAM : [0x200000 - 0x23E7FF] [250 KB]
Please send file using XMODEM and then press ENTER key.
CCCCCCC

( Escape to exit, Space to tag )

[Goto] [Prev] [Show] [Tag] [Untag] [Okay]

CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB0
```

# Example using interrupts (contd...)

- Wait until the process is completed.
- The screen should display how much data has been transferred.

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 14.55:25
+-----[xmodem upload - Press CTRL-C to quit]-----+
Press CTR/Sending uart_interrupt.bin, 239 blocks: Give your local XMODEM
|M receive command now.
CCCCCCCC|Bytes Sent: 30720 BPS:8396
|
|Transfer complete
+-----|READY: press any key to continue...|-----+
| Mi+-----+ndia
+-----+-----+
Bootloader, ver 1.0.0 [ (hdgg@dac_tvm) Tue Dec 15 16:50:32 IST 2020 #135]
|-----+-----+
|-----+-----+-----+-----+-----+-----+-----+
|-----+-----+-----+-----+-----+-----+-----+
ISA : RISC-V [RV32IM]
CPU : VEGA ET1031
SoC : THEJAS32
+-----+-----+-----+-----+-----+-----+-----+
www.vegprocessors.in | vega@dac.in
+-----+-----+-----+-----+-----+-----+-----+
```

Transfer mode : UART XMODEM

IRAM : [0x200000 - 0x23E7FF] [256 KB]

Please send file using XMODEM and then press ENTER key.

CCCCCCCC

- After completing transfer the Program will start to execute.

```
Starting program ...

+-----[ uart0 in interrupt mode ]-----+
Start typing anything on uart0.

VEGA Processors
```

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

Using SDK in IDE (Eclipse)

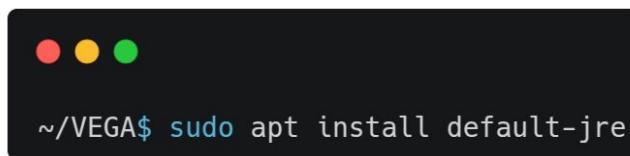
# Using SDK in IDE (Eclipse)

## Installing Eclipse IDE

Eclipse is a Java-based application and it requires a Java runtime environment (JRE) to be installed in order to run.

Install the default OpenJDK package with:

- `sudo apt install default-jre`

A small icon of a terminal window with three colored dots (red, yellow, green) at the top right corner.

```
~/VEGA$ sudo apt install default-jre
```

Download Eclipse IDE installer for Linux 64-bit version from

<https://www.eclipse.org/downloads/>

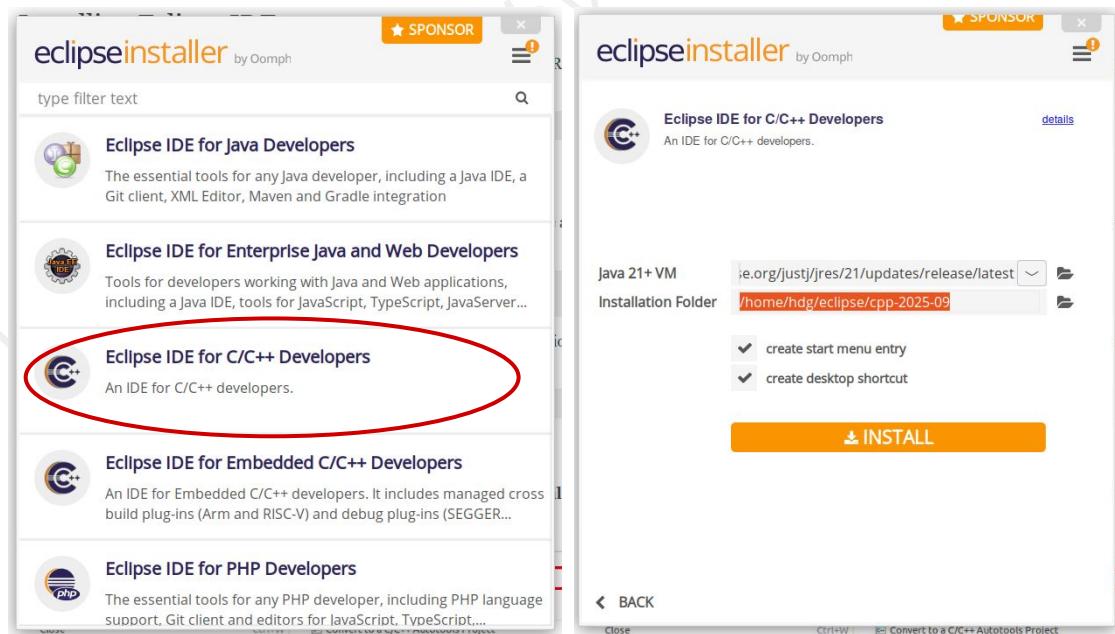
# Using SDK in IDE (Eclipse) (contd...)

## Installing Eclipse IDE

The name of downloaded file will be `eclipse-inst-linux64.tar.gz`

Extract the archive and :

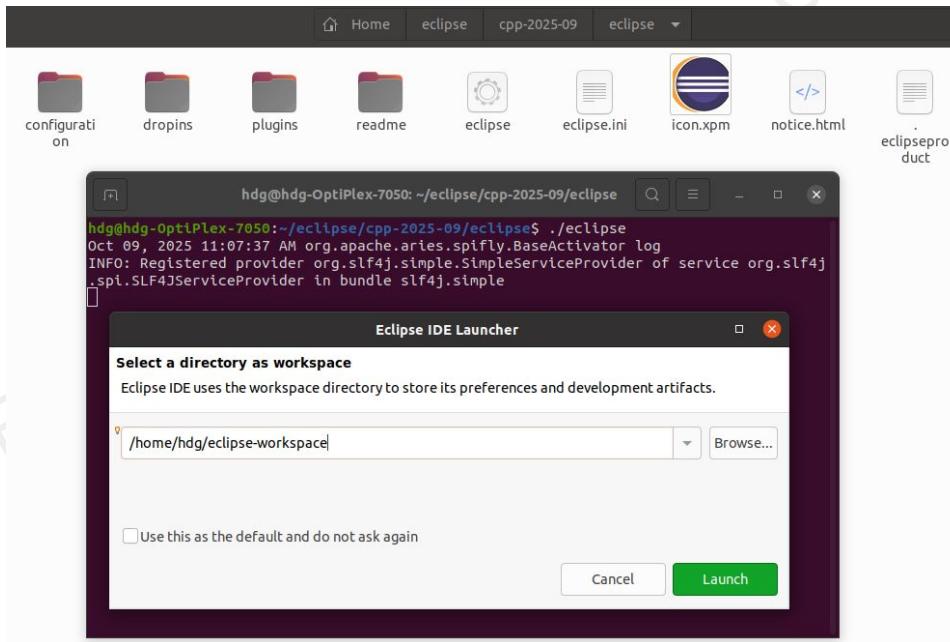
- `cd eclipse-installer`
- `./eclipse-inst`
- Select **Eclipse IDE for C/C++ Developers** and click **INSTALL**.



# Using SDK in IDE (Eclipse) (contd...)

After installation to Open Eclipse go to installation directory

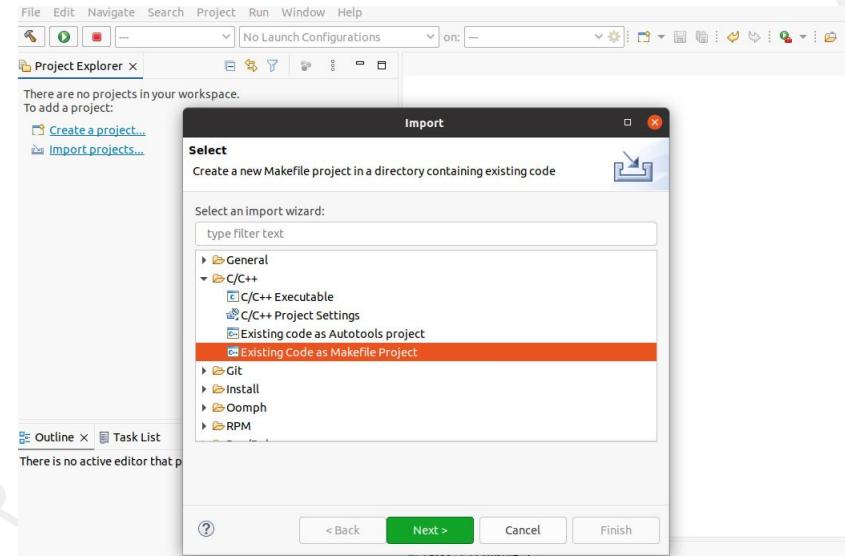
- ./eclipse



# Opening SDK Projects in Eclipse IDE

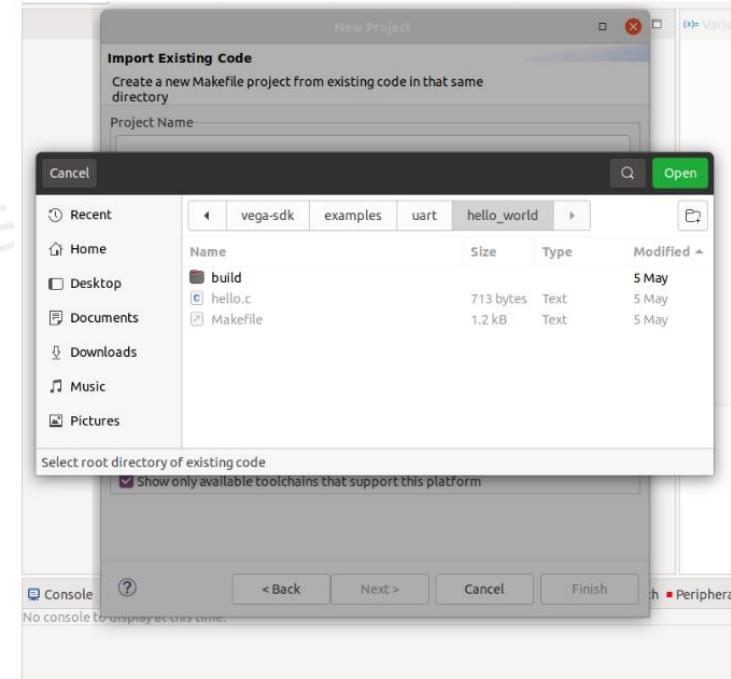
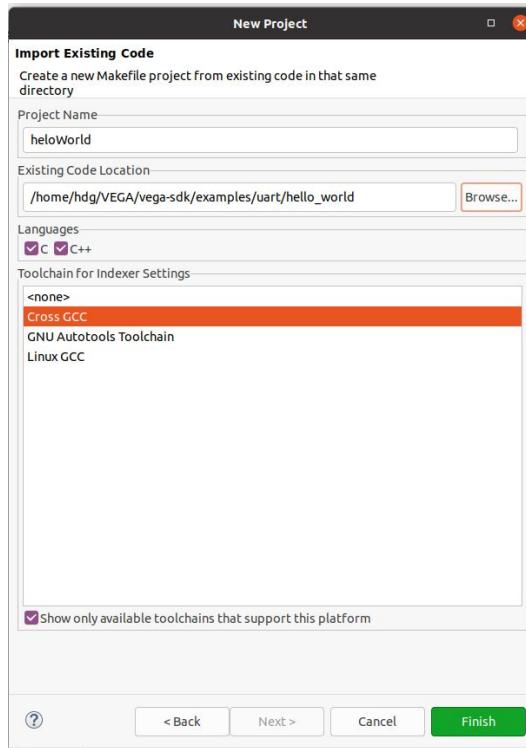


- To open your VEGA SDK project in Eclipse IDE, goto File -> Import -> C/C++ -> Existing Code as Makefile Project (for 2025 version)



# Opening SDK Projects in Eclipse IDE

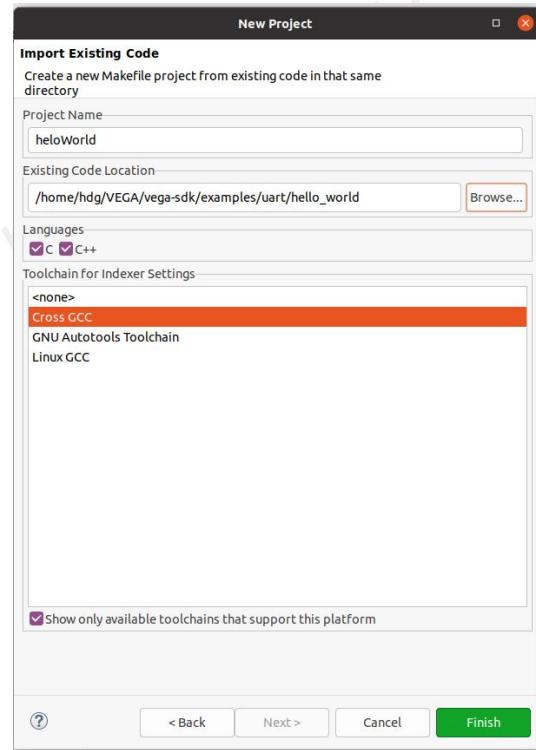
- Browse to existing code location,



# Opening SDK Projects in Eclipse IDE

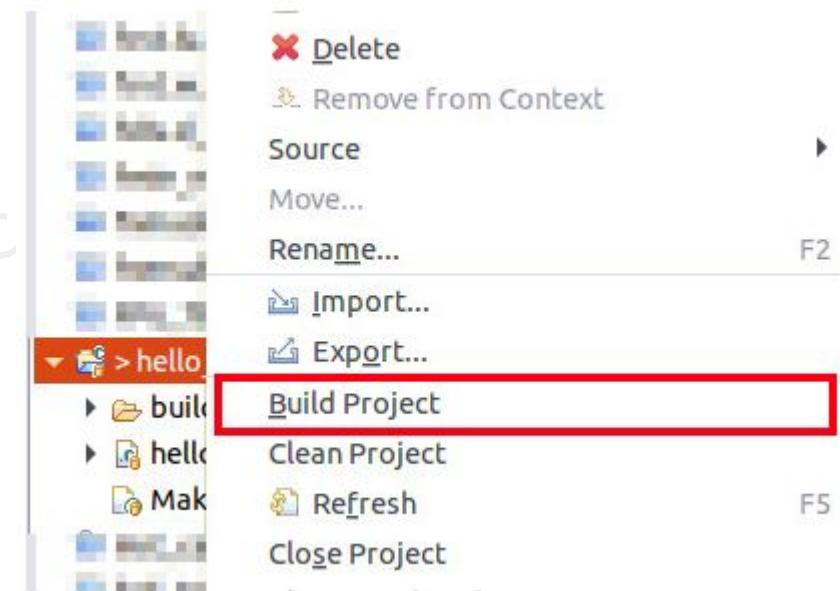
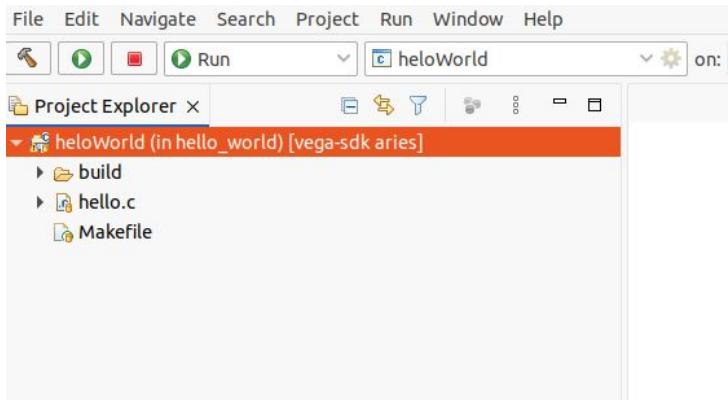


- Select **Cross GCC** option for toolchain indexer and click **Finish** button.



# Building SDK Projects in Eclipse IDE

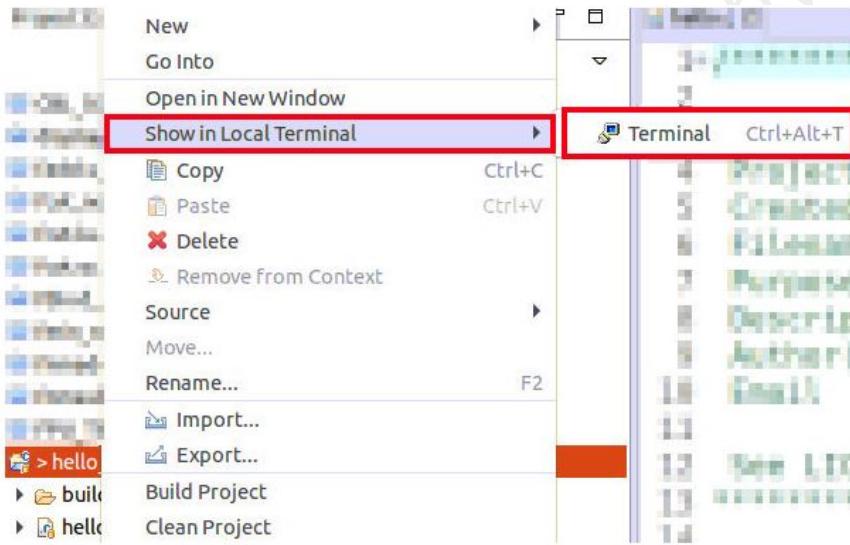
- In the Project Explorer tab right click and select **Build Project** to build the selected project.



- Selecting **Clean Project** will clean the target program build directory.

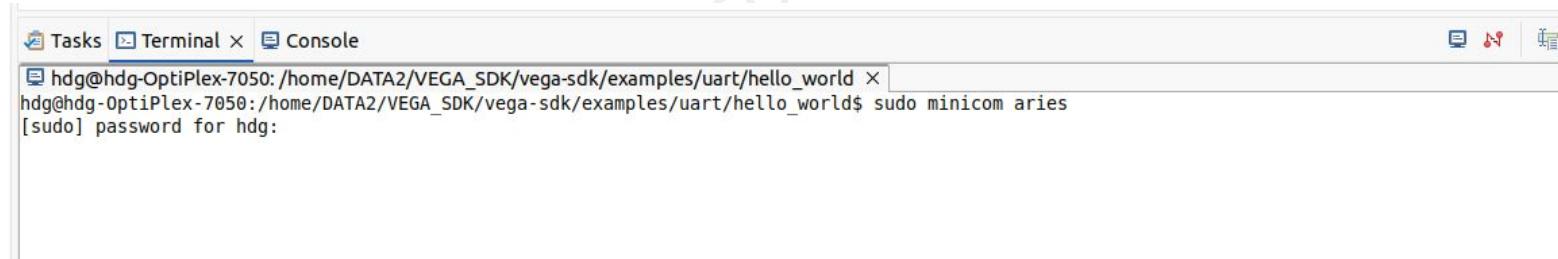
# Running SDK Projects in Eclipse IDE

- In the Project Explorer tab right click and select **Show in Local Terminal** -> **Terminal**.



## Warning

Before giving make upload command do all the steps in [Setting Up Serial Device](#) and [Resetting Target Board](#) subsections.

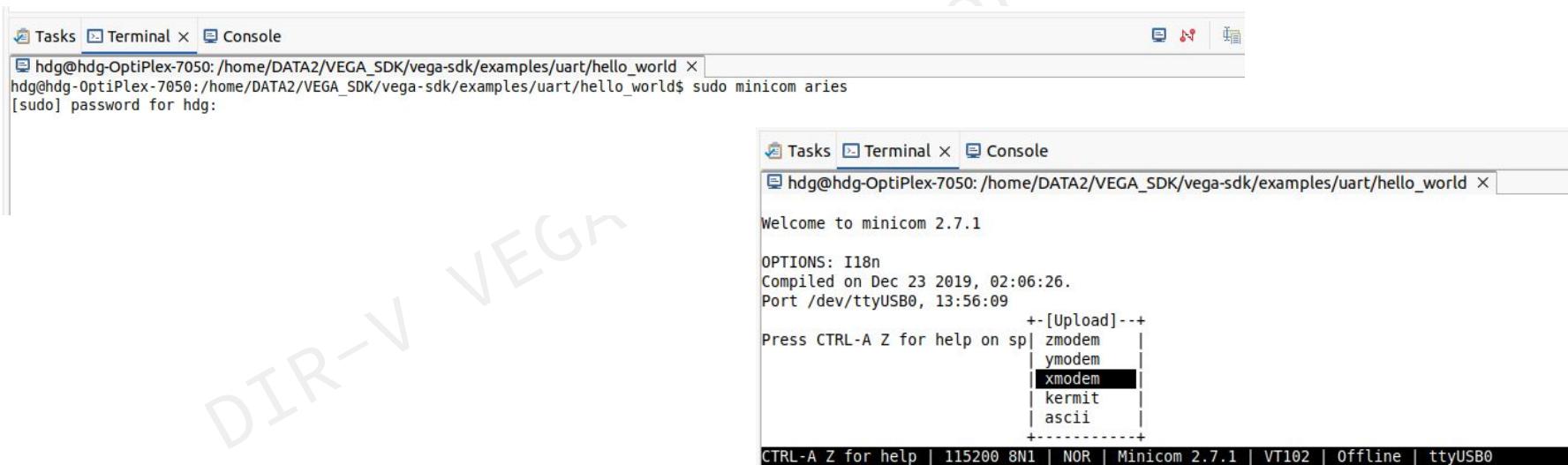


A screenshot of the Eclipse IDE interface, specifically the terminal view. The terminal tab is selected, showing a command-line session. The session starts with the user's home directory and then runs the command 'sudo minicom aries'. A password prompt '[sudo] password for hdg:' is visible at the bottom of the terminal window.

```
hdg@hdg-OptiPlex-7050: /home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world ×
hdg@hdg-OptiPlex-7050:/home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world$ sudo minicom aries
[sudo] password for hdg:
```

## Warning

Before giving make upload command do all the steps in [Setting Up Serial Device](#) and [Resetting Target Board](#) subsections.



Tasks Terminal Console

```
hdg@hdg-OptiPlex-7050: /home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world x
hdg@hdg-OptiPlex-7050: /home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world$ sudo minicom aries
[sudo] password for hdg:
```

Tasks Terminal Console

```
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 13:56:09
+--[Upload]--+
Press CTRL-A Z for help on sp| zmodem |
| ymodem |
| xmodem |
| kermit |
| ascii |
+-----+
```

CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB0

# Programming using minicom in Eclipse IDE (contd...)



## Warning

Before giving make upload command do all the steps in [Setting Up Serial Device](#) and [Resetting Target Board](#) subsections.

The screenshot shows the Eclipse IDE interface with two terminal windows and a code editor.

**Terminal Window 1:** Minicom 2.7.1 session.

```
Welcome to minicom 2.7.1
OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB0, 13:56:09
Press CTRL-A Z for help on special keys
+-----[Select a file for upload]-----+
|Directory: /home/hdg/VEGA_SDK/vega-sdk/bin
| [...]
| custom.project.bin
| flasher.bin
| hello.bin
| max485_demo.bin
| ( Escape to exit, Space to tag )
+-----+
[Goto] [Prev] [Show] [Tag] [Untag] [Okay]

CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB0
```

**Terminal Window 2:** Another minicom session or a different terminal window.

```
hdg@hdg-OptiPlex-7050:/home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world
IRAM : [0x200000 - 0x23E7FF] [250 KB]

Please send file using XMODEM and then press ENTER key.
CCCCCCC
Starting program ...

Hello World
EXIT
```

## Warning

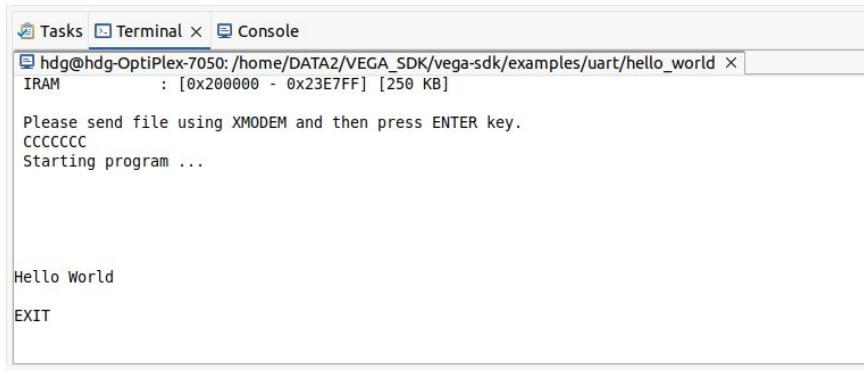
Before giving make upload command do all the steps in [Setting Up Serial Device](#) and [Resetting Target Board](#) subsections.

```
hdg@hdg-OptiPlex-7050:~/home/DATA2/VEGA/vega-sdk/examples/uart/hello_world$ make upload
TARGET: THEJAS32 Hardware
C Files : [./hello.c]
CPP Files : []
Assembly Files : []
/home/DATA2/VEGA/vega-tools/toolchain/bin
-march=rv32ima zicsr -mabi=ilp32 -mcmodel=medany
Changing mode to XMODEM ...Waiting for receiver ping ...done.
Sending /home/DATA2/VEGA/vega-sdk/examples/uart/hello_world/build/hello.bin .....done.
# @echo "Please connect the aries board to PC and enter your password to open minicom";
sudo minicom trisul32
[sudo] password for hdg:
```

# Programming using make upload in Eclipse IDE

In Terminal Tab give command

- make upload



The screenshot shows the Eclipse IDE interface with the "Terminal" tab selected. The terminal window displays the following text:

```
Tasks Terminal Console
hdg@hdg-OptiPlex-7050: /home/DATA2/VEGA_SDK/vega-sdk/examples/uart/hello_world >
IRAM : [0x200000 - 0x23E7FF] [250 KB]

Please send file using XMODEM and then press ENTER key.
CCCCCCC
Starting program ...

Hello World
EXIT
```

**Note:** Reset the board first, then run the “make upload” command before switching to Ethernet mode; otherwise, you won’t get the correct output.

# DIR-V GRAND CHALLENGE

## VEGA Processor Tutorial

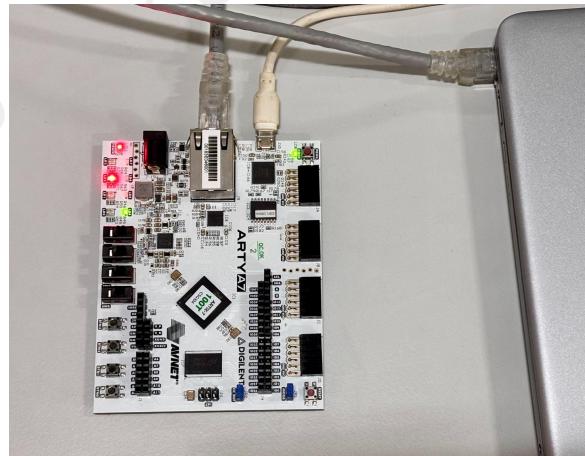
Uploading Using Ethernet

# Uploading Using Ethernet

Make sure that you connected ethernet cable to board and board is reset before you upload program.

## Warning

Before giving make upload command do all the steps in [Setting Up Serial Device](#) and [Resetting Target Board](#) subsections.



# Uploading Using Ethernet (contd...)



- After resetting the board, ensure it is in Ethernet mode.
- The board will then wait for incoming data and display the message “waiting for data...” as shown.

```
/VEGA$ sudo minicom trisul32
[sudo] password for hdg:
Welcome to minicom 2.7.1

OPTIONS: I18n
Compiled on Dec 23 2019, 02:06:26.
Port /dev/ttyUSB1, 11:23:24

Press CTRL-A Z for help on special keys

+-----+
| VEGA Series of Microprocessors Developed By C-DAC, INDIA           |
| Microprocessor Development Programme, Funded by MeitY, Govt. of India |
| Swadeshi Microprocessor Challenge - Innovate Solutions for #AtmaNirbhariBharat |
+-----+
| Bootloader, ver 1.0.0 [ (hdg@cdac_tvm) Mon Jun 30 11:01:32 IST 2025 #160] |
| | ISA : RISC-V [RV32IMA] |
| | CPU : VEGA AT1051 |
| | SoC : TRISUL32 |
+-----+
| www.vegaprocessors.in | vega@cdac.in |
+-----+
Press any key to choose the xmodem transfer

The ethernet mode will be executed automatically in 00s.

Transfer mode : Ethernet

Memory       : DDR2 [0x80000000 - 0xffffffff] [256 MB]

MAC: aa:bb:cc:dd:ee:ff

Ethernet Initialised @ 100Mbps

Waiting for data....
```

# Uploading Using Ethernet

(contd...)

- You could upload program to target board using `make` command.
- To upload program use :  
`make ethupload`
- If the file was uploaded successfully, then you will see the following message in terminal.

```
~/VEGA/vega-sdk/examples/uart/hello_world$ make ethupload
TARGET: THEJASS2 Hardware
C Files : [./hello.c]
CPP Files : []
Assembly Files : []
/home/hdg/VEGA/vega-tools/toolchain/bin
-march=rv32ima_zicsr -mabi=ilp32 -mcmodel=medany
sudo ./ether enp2s0 /home/hdg/VEGA/vega-sdk/examples/uart/hello_world/build/hello.bin
[sudo] password for hdg:
No of files :1
address 1 : 0x80000000 [ /home/hdg/VEGA/vega-sdk/examples/uart/hello_world/build/hello.bin ]
entry : 0x80000000
entry : 0x20000
Mac : aabbccddeeff
Total files :1
File : /home/hdg/VEGA/vega-sdk/examples/uart/hello_world/build/hello.bin
Read count 26092
File checksum: 246427
```

# Uploading Using Ethernet (contd...)

## Sample File Upload Output

```
The ethernet mode will be executed automatically in 00s.  
Transfer mode : Ethernet  
Memory : DDR2 [0x80000000 - 0x8fffffff] [256 MB]  
MAC: aa:bb:cc:dd:ee:ff  
Ethernet Initialised @ 100Mbps  
Waiting for data.....  
Download settings received  
Total files : 1  
File 1 address : 80000000  
Entry point : 80000000  
Waiting for file 1  
File received successfully  
Total bytes : 26092 ,[80000000] - [800065ec]  
  
Hello World  
Hello World
```

# THANK YOU

[vega@cdac.in](mailto:vega@cdac.in)

**Mob:9037569219**